

SIGNIFICANCE OF HASH CODE IN MODERN COMPUTING

Ayushi Rai¹, Ananya Srivastava², Mukta Gupta³, Ashish Jaiswal⁴
B. Tech. Computer Science & Engineering
Institute of Technology and Management GIDA, Gorakhpur

Abstract: Hashing is a technique used to improve searching. Hash function is a function, which take a group of character and map it to certain length of character and convert it to some other form of string. Hash function generates unique code for each unique input. Keys are stored in the table at the addresses generated by the hash function. Table has a starting and ending address. It is expected from the hash function to generate the code of the valid limit. This is because to use the available spaces efficiently. A hash function is said to be collision-free if it generates the unique codes for each unique inputs otherwise collision will take place. These two properties make the designing of hash function complex. Hash functions having the above mentioned properties are said to be perfect hash function. In spite of these challenges, hashing is widely used in the areas of Database System, Symbol Table, Data Dictionary, Cryptography and Network Security, Network Processing Algorithm, Browser Caches and others. This paper present a descriptive study of Hash function and its application in various fields like security, storage, medical and retrieval from database. This paper also describe about various cryptography hash function such as SHA, MD, RIPEMD and non cryptography hash functions such as Pearson, Javahash code, City, Bemstein and their variant use for various purpose. This paper also discuss about the problem associated with the hash function like collision and birthday paradox and how to recover from those problems. This paper is shedding light on various aspects where hashing appears as a major problem solving technique.

Keywords: Hash functions, collision free hash functions, perfect hash function, database systems, cryptography and network security.

I. INTRODUCTION

In hash function a group of characters are taken and this group of characters are called a key and then it maps it to a value of a certain length which is called a hash value or hash. The original string of characters is the representative of the hash value, but normally it is smaller than the original. In database, hashing is used to enable items to be retrieved more quickly. In digital signature, hashing can also be used in the encryption and decryption. The digital signature is transformed by the hash function, then both the signature and hash value are sent to the receiver. The same hash function is used by the receiver to generate the hash value and then compares it to the received message. If the hash values are the same then the message was transmitted without errors. There is an example of hash function called folding. In folding, an original value is taken and it is divided into

several parts, after that, add all the parts and use the last four remaining digits as the hashed value or key. There is another example of hash function is called digital rearrangement. In digital rearrangement, take the digit in certain positions of the original value such as the second and fourth numbers and reverse their order. It then uses the number left as the hash value. It is almost impossible to determine the original number based on a hashed value unless the algorithm that was used is known to us. A hash table is collection of items which are stored in such a way as to make it easier to find them later. In hash table each position is called a slot. Slot can hold an item and it is named by an integer value starting from 0. The hash function will take any item in the collection and return an integer in the range of slot names, between 0 and m-1. In many situations hash tables turn to be more efficient than search trees or any other table lookup structure.

II. TYPES OF HASH FUNCTION

There are two types of hash function. First is cryptographic hash function and second is non-cryptographic hash function.

A. Cryptographic Hash Function

There are following types of cryptographic hash function are discussed below.

- *Eilliptic Curve Only Hash:* The MuHASH hash algorithm is used by ECOH as based function. MuHASH is ineffective for practical use and changes had to be made. ECOH uses a padding function. It divides the message in the blocks. It has a simple and clear design, it support the Intel AES instruction set. ECOH algorithms have our versions, ECOH-224, ECOH-256, ECOH-384 and ECOH-512. The number 224, 256, 384 are size of the message digest. The ECOH algorithm is about thousand times slower than SHA-1 [1].
- *Fast Syndrome-based Hash Function:* The Fast Syndrome-based hash Function (FSB) developed in 2003 by Matthieu Finiasz, Daniel Augot, Nicolas Sendrier. Not like most other cryptographic hash functions in use today, FSB have some point be proven to be secure. All versions of FSB claim provable security; some beginning versions were eventually broken.FSB is slower than traditional hash functions and it also uses lot of memory, so it is impractical in memory constrained area [2].
- *Has-160:* Korean KCDSA digital signature algorithm uses HAS-160 mainly it designed for this. It is derived from SHA, with various changes proposed to increase its security. It does generate a 160-bit output.HAS-160 is used in the equal manner

as SHA-1. It split input in chunks of 512 bits and pads the last block. By processing the input block in turn a digest function updates the intermediary hash value. The message digest algorithm contains 80 rounds.

- **HAVAL:** HAVAL is a cryptographic hash function. Different lengths hashes produced by HAVAL. HAVAL can produce hashes in length of 128 bits, 16 bits, 192 bits, 224 bits and 256 bits. HAVAL also has provides the elasticity to modify the number of passes message blocks are processed [16].
- **MD2:** The cryptographic hash function MD2 is developed by Ronald Rivest in 1989. MD2 is no longer considered secure, even as of 2014, it remains use in public infrastructures as piece of certificates generated with MD2 and RSA [4].
- **MD4:** The MD4 Message-Digest Algorithm is developed in 1990 by Ronald Rivest. In this hash function input is random size and output is set 128 bit. The algorithm has carried in afterward designs, like MD5, SHA-1 and RIPEMD algorithms. The security is strictly compromised for MD4. The initial full collision attack against MD4 was introduced in 1995 and more than a few attacks have been published since then [3].
- **MD5:** Ronald Rivest designed MD5 in 1991. It replaces a previous hash function, MD4 and MD5 message-digest algorithm is used broadly. It producing a 128-bit (16-bit) hash value, usually expressed in text format as a 32 digest hexadecimal number. MD5 used in a wide range cryptographic application, and is also used to confirm data integrity [3].
- **MD6:** It uses a Merkle tree-like construction to allow for vast parallel computation of hashes for very lengthy input [5].
- **RIPEMD:** RIPEMD is stand for Race Integrity Primitives Evaluation Message Digest is developed by Hans Dobbertin, Antoon Bosselaers and Bart Preneel. RIPEMD is similar in performance to the SHA-1. RIPEMD-160 is an improved version of RIPEMD, and the most general version in the family [3].
- **SHA:** SHA is a type of Cryptographic hash function. It was published by NIST (National Institute of Standards & Technology). It consists of various types like SHA1, SHA2, SHA3.
- **SHA -1:** It produces a 20 byte hash value (192 bits) & it is 40 digit long. It came in 1995. SHA -1 provides better resistance than SHA-0. It is widely used in applications like Cryptography & Data Integrity [6].
- **SHA-2:** It gives hash values of 224, 256, 384 or 512 bits i.e. SHA-224, SHA-256, SHA-384, SHA-512. It came up in 2001. Several defects in SHA-1 gave need to produce another hash function (SHA-3).

SHA-256 AND SHA 512 just differs in number of round and they are computed by using 32 and 64 bit.

- **SHA-3:** It does not come up with any strong reason after SHA-1 and SHA-2 but with any possibility that it may face attack so it was introduced [5].

B. Non-Cryptographic Hash Function

There are following types of non-cryptographic hash function are discussed below.

- **Pearson Hashing:** Pearson hashing is a hash function used for fast execution on processors with 8-bit registers. Input consisting of any number of bytes and it produces as output a single byte. Its execution requires only a few instructions [7]. It offers these benefits:

It is very easy.

Its execution is very fast on resource-limited processors.

There is no plain class of inputs for which collisions are especially likely.

- **Jenkins Hash Function:** The Jenkins hash functions designed by Bob Jenkins. Jenkins hash function is a set of hash functions for multi-byte keys. They can be used also as checksums to discover accidental data corruption or detect identical records in a database. The first paper was formally published in 1997 [8].

- **Fowler–Noll–Vo Hash Function:** Fowler–Noll–Vo is created by Glenn Fowler, London Curt Null, and Phong Vo. The beginning of the FNV hash algorithm was taken from any plan sent as reviewer comments to the IEEE POSIX P1003.2 committee headed by Glenn Fowler and Phong Vo in 1991. The recent versions are FNV-1 and FNV-1a, which give a means of creating non-zero FNV offset basis. FNV now comes in 32-, 64-, 128-, 256-, 512-, and 1024-bit flavours.

- **Java hashCode():** All class implicitly or explicitly provides a hashCode() method in the Java programming language, which digests the data stored in an instance of the class into a particular hash value [9].

- **Zobrist Hashing:** Zobrist hashing as well known as Zobrist keys or Zobrist signatures. It is used in computer programs that play abstract board games, such as chess, to implement transposition tables, a particular kind of hash table that is indexed by a board position and used to avoid analyzing the similar position more than once. Zobrist hashing is named for its discoverer, Albert Lindsey Zobrist [10].

- **Bernsteins Hash:** It was designed by Dan Bernstein and it gives 32 bit hash code. It does not give good performance in Avalanche and permutation of internal states. It gives good results for short keys but it cause excessive collision.

- **Murmur Hash:** It was designed by Austin Appleby in 2008. It has other variants which are released in public domain. It's variants are MURMUR1 which is obsolete, MURMUR2 yield 32 bits or 64 bits value, MURMUR64A is designed for 64 bit processor and MURMUR64B is designed for 32 bit processor. It performs well in random distribution of regular key when compared with other hash functions. It

has been implemented in various languages like C++, C, Python, Perl, Ruby, Php, Java etc. It is also adopted in many open source projects [11]. It's some features are-

- Very Simple.
 - Good Distribution.
 - Good Avalanche behaviour.
 - Good collision resistance.
 - Good performance
1. Spooky Hash: It is a public domain non cryptographic hash function. It produces 32, 64 or 128 bit hash values for an input of any length. It was named as spooky hash because it was released on Halloween. It is very fast with 1 byte per cycle for short keys and with 3 bytes per cycle for long keys. It could cause resistance.
 2. City Hash: It came in 2010. It is used for fast hashing of string. It's variants are 32, 64, 128, 256 bit variants. It is partly based on Murmur Hash. It does not maintain compatibility with previous version so it must not be used for longer storage and if used then must not be upgraded. It gives good state diffusion. City hash 64 is 1.5 times faster than Murmur2. For longer string City Hash is 1.3 to 1.6 times faster than Murmur2.
 3. xx Hash: It is a very fast Non cryptographic hash algorithm. It's speed is close to RAM limit. It's hash code length is either 32 bit or 64 bits.

III. PERFECT HASH FUNCTION

A perfect hash function is defines that it maps the set of actual key values to the table without any collisions. A minimal perfect hash function does so using a table that has only as many slots as there are key values to be hashed. It is possible to construct a specialized hash function that is perfect, perhaps even minimal perfect if the set of keys is known in advance.

A. Minimal Collision Resistant

Minimal collision resistant is a drawback of perfect hash function. A function family $\{f_a\}_{a \in A}$ is said to be collision-resistant if given a uniformly chosen $a \in A$, it is infeasible to find elements $x_1 \neq x_2$ so that $f_a(x_1) = f_a(x_2)$. Collision-resistant hash functions are one of the most widely-employed cryptographic primitives. Their applications include integrity checking, user and message authentication, commitment protocols, and more [12].

IV. CICHELLI'S METHOD

The Cichelli's method is used when it is necessary to hash a relatively small collection of keys, such as the set of reserved words for a programming language.

The basic formula is: $h(S) = S.length() + g(S[0]) + g(S[S.length()-1])$ Here using Cichelli's algorithm $g()$ is constructed so that $h()$ will return a different hash value for each word in the set.

V. THE BIRTHDAY PARADOX: A PROBABILITY PROBLEM

The birthday problem is defines that it is a probability problem that deals with the probability of at least one pair of people in a group of n people that share the same birthday. Now here, a question arises that "How many people are required in which at least two people in the group share the same birthday? To many, this might seem like a large number of people are required. However, this is incorrect, as only 23 people are required to have a probability of around 50.7%.. However, instead, in this birthday problem, every birthday is randomized [13].

A. Solving the Problem

To solve this problem as, one must find the probability of the complement by subtracting this probability from one, one could find the probability of at least one pair having the same birthday. For example, the probability of 23 people having at least one of the same birthday as follows:

$$\begin{aligned}
 P(A) &= 1 - P(\bar{A}) \\
 P(\text{At least one same birthday}) &= 1 - P(\text{no same birthday}) \\
 &= 1 - (365/365 * (365-1)/365 * \dots * (365-23)/365) \\
 &\sim 0.892
 \end{aligned}$$

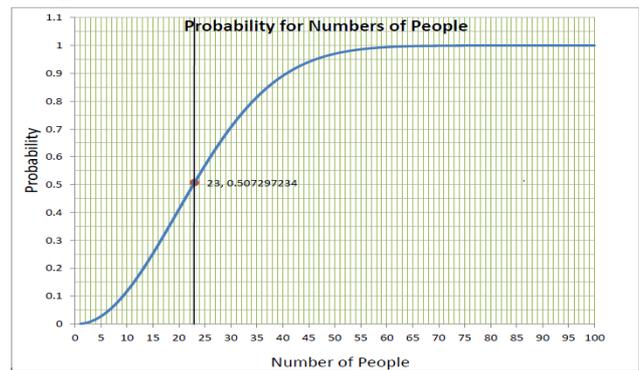


Figure 1: Probability for numbers of people having at least one of the same birthdays.

B. Application of Birthday Paradox

The birthday problem is used in the birthday attacks. It is a brute-force cryptographic attack against hash function problems. These hash functions are either well-defined procedures or mathematical functions which convert large amounts of data into a single-integer datum, called a hash value, hash code, hash sum, or checksum. This is done in order to speed up the looking up of items in a database. However, there are a limited amount of hashes. So, there can be many hash collisions, it is the base on which the birthday attack operates. In case of information security purposes, cryptographic hash functions exist where the data is taken and converted into a fixed-size bit string. It must be easy to compute a hash value for a message for an ideal cryptographic hash function, but to create a message with a given hash is infeasible, infeasible to modify a message without changing the hash and infeasible to find different messages with the same hash. Any small changes in the input will lead to a strongly different output [13].

VI. APPLICATIONS OF HASH FUNCTION

A. Parallel Collision Search

In this we first review known methods for collision search, and also describe how to efficiently parallelize this task. The goal in collision search is to find two different inputs that produce the same output and also take a given function f . The function f is chosen so that finding a collision serves some cryptanalytic purpose. A method for finding a collision is to select distinct inputs x_i for $i = 1, 2, \dots, k$ and check for a collision in the $f(x_i)$ values. Let the cardinality of the range of f be the n . The probability that no collision is found after selecting k inputs is $(1-1/n)(1-2/n)\dots(1-(k-1)/n) \approx e^{-k^2/(2n)}$ for large n and $k = O(\sqrt{n})$. The large memory requirements can be eliminated using Pollard's rho method. This method involves taking a pseudo-random walk through some finite set S .

Finding Real Collisions in Hash Functions: We apply the parallel collision search technique to finding real collisions in hash functions. We first review how hash functions are typically used in conjunction with digital signatures, and the classic attack of Yuval. Then we can apply parallel collision search to extend this attack allowing parallelization and reducing memory requirements. Consider as an example the impact on the MD5 hash function. Hash functions are designed to take a message consisting of an arbitrary number of binary bits and map it to a fixed size output called a hash result. Let $H: M \rightarrow R$ is such a hash function. Typically, hash functions are constructed from a function $h: B^l \rightarrow R$ which takes a fixed size block of message bits together with an intermediate hash result and produces a new intermediate hash result. A given message $m \in M$ is typically padded to a multiple of the block size and split into blocks $m_1, \dots, m_l \in B^l$. The padding often includes a field which indicates the number of bits in the original message. Beginning with some constant $r_0 \in R$, the sequence $r_i = h(m_i, r_{i-1})$ is computed for $i = 1, \dots, l$, and r_l is the hash result for message m . Hash functions are commonly used in connection with digital signatures. Instead of direct signing a message, first the message is hashed and then sign the hash result. A digital signature could be moved from one message to the other for cryptographic security it must be computationally infeasible to find two messages that hash to the same value. Now suppose we have a message m that we would like our adversary to digitally sign, but he is not willing to do so [17].

B. Distributed Anonymous Information Storage and Retrieval System

Computer networks are growing day by day in importance as a medium for the information exchange and storage. However, little bit privacy afford by the latest systems to their users, and generally store any given data item in only one or a few fixed places, creating a central point of failure. Now a days, individuals want to protect the privacy of their authorship or readership of various types of sentient information and undesirability of central points of failure which can be attacked by opponents wishing to remove data from the system or simply overloaded by too much interest that's why systems offering more security and trustworthy

are needed. So, there was development of freenet, a distributed information storage and retrieval system designed to address these concerns of privacy and accessibility. The system operates, as a independent distributed file system across many individual computers as a location that allows files to be requested, inserted and stored.

There are five main design goals:

- Namelessness of information for both producers and consumers
- Deniability for stores of information
- Opposition to attempts by third parties to refuse access to information
- Efficient routing of information and dynamic storage
- All network functions are decentralized

Efficient service is provided without resorting to broadcast searches or centralized location indexes the system is designed to respond adaptively to usage patterns, duplicating, transparently moving and deleting files as necessary. It is not mean to guarantee permanent file storage, although it is thought that's much nodes will join with enough storage capacity that most files will be able to remain indefinitely. It is transport independent because the system operates at the application layer and assumes the existence of a secure transport layer. For general network usage, it does not look for providing namelessness only for freenet file transactions [21].

C. Use in Medical Area

Area in medical where hashing is used are discussed below:
Comparison of Medical Images Using Geometric Hashing: We can register 3D Medical Image using Geometric Hashing. Here registering mean determining the rotation and translation displacement that aligns two images to obtain the best overlap of anatomical structures of the 2 images. Image registration is as a fundamental problem in Medical Imaging. Comparing medical images is very useful, and sometimes necessary, for diagnosis, patient follow up, treatment planning and surgical planning. Today a variety of medical image measures are available, such as conventional X-ray, X-ray Fluoroscopy, X-ray Computed Tomography (CT), Magnetic Resonance Imagery (MRI), Positron Emission Tomography (PET) Ultrasound (resulting images are sometimes called Sonograms), Single Photon Emission Computed Tomography (SPECT). Without an accurate geometric overlaps, it is risky to compare or fuse images corresponding to the same anatomy. There are two main categories are intensity or pixel based techniques and feature based techniques. In Intensity based techniques, a pixel is matched between different images and attempt to optimize registration parameters in order to maximize this measure. Feature based techniques is based on pre-processing step to identify various anatomical structures in both images. Then the structures are then registered and the transformation that was superimposed between the structures is applied to the underlying images. The main reason for using Geometric Hashing over other matching methods is its useful computational complexity. Geometrical Hashing is very well

suitable when one wishes to register a shape against a database of shapes, as the complexity of the algorithm is not affected by the size of the database. We can design a very efficient registration algorithm based on the idea of Geometrical Hashing if the rigid are found and their information are stored developing a special purpose algorithm for curves can result in increased efficiency and speed [19].

Watermarking Technique for Authentication of Content: The watermark is a technique in which message authentication codes, patient information and hospital logo are computed using hash function and embedded in RONI. Medical image contains region of interest (ROI) and region of non interest (RONI). ROI consist the important information from diagnosis point of view and it must be stored without any distortion. Watermarking contains the information of ROI and with help of this technique distortion of ROI is prevented. The Watermarking allows permanent association of image content with proofs of its reliability by modifying the image pixel values. There are two types of watermarking first the robust water marking and second the fragile water marking. Robust watermarks are those which are difficult to remove from the digital information. This is basically used for copyright applications because of intentional or incidental distortions like compression, scaling, cropping, filtering, A/D or D/A conversion, etc. Fragile watermarks are those that are destroyed easily by tampering or modifying the watermarked content so the absence of watermark to the previously watermarked content points to the conclusion that data has been tampered. There is a blind fragile watermarking scheme that does not require the original host image during the extraction of watermark. LSB is a simple embedding technique with a high embedding capacity and small embedding distortions. The LSBs of image are generally the noise caused by the imaging device. These bits can be used for secret message embedding without greatly disturbing the image overlook. This scheme is useful for both the purposes of medical image authentication and hiding electronic patient record. It is technique of inserting electronic patient record (EPR) data in medical images. EPR data contain text file and graphs. Text file is the report from the radiology department of the hospital about the patient and graphs are ECG or EEG. The LSB technique implemented in spatial domain . The ASCII characters in EPR data are encrypted using Rijndael Algorithm before hiding in medical images to improve the security of the data. How to generate water mark, in order to generate the watermark the following steps are implemented [20]:

- Read the logo image of the hospital.
- Find global threshold of logo image using Otsu's method.
- Convert an intensity logo image into a binary image using this threshold and then apply the BCH encoding to encode it. Let it be vector v1.
- Read the text file and convert each character of text file into its corresponding ASCII code.
- Then Convert each ASCII code into its corresponding binary code. Apply the BCH

encoding again to this vector and call it v2.

- Set LSBs of all the pixels in the input image to zero and find the Hash function of this image. Hash function will give 32 characters string. Convert it into binary string and then apply the BCH encoding and call it v3.
- Now concatenate all the watermarks v1, v2& v3and call it V that is the resultant watermark to be embedded into the host image.

D. Security

There are many type of application in hash function but security is the most important application. The security of several information and maintained hash function in hash function is essential. In hash function Security is a field of applications that provides privacy, authentication and confidentiality to users. An important sub-field is that of secure communication and allowing confidential communication between different parties, such that no illegal party has right to use to content of the messages. Security for information has three components is confidentiality, integrity and availability. The hash functions secure communication systems by using the messages digits. For security use digital signatures, message-digest algorithm 5 (MD5), MD4, HAVAL, FORK-256, SHA-family, RIPEMD-family are used .Various hash standards such as SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512 are used to measure the Secure Hash Standards (SHS) levels. A hash function takes a input of any length and gives an output of fixed length which acts as a type of signature for the data provided. Using digital signatures verification of the authentication of the data, in this process hash function compare by the sender and the receiver to know it is genuine or not. The digital signature is the electronic analog. If any changes occurred in data will affect the hash code which is sent with the data. Digital signature provides a number of security services which are Message Authentication, Message Integrity, and Confidentiality [15].

VII. CONCLUSION

This is a survey paper on the hash function and we have discussed the varied application of hash functions in the field of security, medical storage, retrieval from database and the problem associated with hash functions like birthday paradox, collision and discussed the solution. To enhance the working of hash function in their respective field we have discussed the various types of hash function with their pros and cons in the paper. Various researches to get most efficient hash function are still taking place. With its multi use it will enhance its application and working in upcoming technologies.

REFERENCES

- [1] Daniel R. L. Brown, Matt Campagna, Rene Struik (2008). "ECOH: the Elliptic Curve Only Hash".
- [2] Augot, D.; Finiasz, M.; Sendrier, N. (2003), "A fast provably secure cryptographic hash function."
- [3] Collisions for Hash Functions MD4, MD5,

- HAVAL-128 and RIPEMD.
- [4] Burt Kaliski, RFC 1319-MD2 Message Digest Algorithm, April 1992.
- [5] Ronald L. Rivest. "The MD6 hash function A proposal to NIST for SHA3".
- [6] Xiaoyun Wang, Yiqun Lisa Yia and Hongbo Yu, Finding Collisions in the Full SHA-1, Crypto 2005 MIT.edu
- [7] Peter K. Pearson, "Fast Hashing of Variable Length Text Strings".
- [8] Jenkins, Bob(c. 2006). "A hash functions for hash Table lookup". Retrieved April 16, 2009.
- [9] Bloch, Joshua (2008) "Always Override hashCode when you override equals" *Effective Java(2 ed.)*, Addison-Wesley.
- [10] Albert Lindsey Zobrist, *A New Hashing Method with Application for Game Playing*, Tech. Rep. 88, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, (1969).
- [11] "Murmurhash3 PHP extension". Murmur.vaizard.org. Retrieved 13 January 2010.
- [12] Chris Peikerty Alon Rosenz "Efficient Collision-Resistant Hashing from Worst-Case Assumptions on Cyclic Lattices" December 11, 2005.
- [13] "The Birthday Paradox: A Simple and Surprising Probability Problem", Jared Sun 3 August, 2011.
- [14] Imad Fakhri Al-shaikhli Mohammad A. Alahmad Khanssaa Munthir "Hash Function of Finalist SHA-3: Analysis Study" International Journal of Advanced Computer Science and Information Technology (IJACSIT) Vol. 2, No. 2, April 2013, Page: 1-12, ISSN: 2296-1739
- [15] The Exact Security of Digital Signatures- How to Sign with RSA and Rabin Mihir Bellare and Phillip Rogaway.
- [16] Appeared in "Advances in Cryptology — AUSCRYPT'92," Lecture Notes in Computer Science, Vol.718, pp.83-104, Springer-Verlag, 1993. HAVAL — A One-Way Hashing Algorithm with Variable Length of Output (Extended Abstract) YuliangZheng , Josef Pieprzyk and Jennifer Seberry Department of Computer Science University of Wollongong, Wollongong, NSW 2522, Australia.
- [17] "Parallel Collision Search with Application to Hash Functions and Discrete Logarithms" *Paul C. van Oorschot and Michael J. Wiener*, Bell-Northern Research, P.O. Box 3511 Station C, Ottawa, Ontario, K1Y 4H7, Canada, 1994 August 17.
- [18] An extended abstract of this paper appears in Advances in Cryptology {EUROCRYPT '04, Lecture Notes in Computer Science Vol. 3027, C. Cachin and J. Camenisch eds., Springer-Verlag, 2004. This is the full version. Hash Function Balance and its Impact on Birthday AttacksMihirBellareTadayoshiKohnoy May 2004.
- [19] "Medical Image Registration using Geometric Hashing", Xavier Pennec.
- [20] "Watermarking of Chest CT Scan Medical Images for Content Authentication", Nisar A. Memon, S.A.M. Gilani,, and Asad Ali.
- [21] Freenet_ A Distributed Anonymous Information Storage and Retrieval System Ian Clarke 124C Lyham Road, Clapham Park London SW2 5QA United Kingdom Oskar Sandberg Morbydalen 12 18252Stockholm Sweden Brandon Wiley USA Theodore W. Hong.